UIUCDCS-R-79-962

UILU-ENG 79 1708
COO-2383-0057

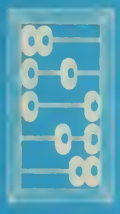
# INITIAL VALUE PROBLEMS: PRACTICAL THEORETICAL DEVELOPMENTS

by

C. W. Gear



March 1979



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

UIUCDCS-R-79-962

# INITIAL VALUE PROBLEMS: PRACTICAL THEORETICAL DEVELOPMENTS

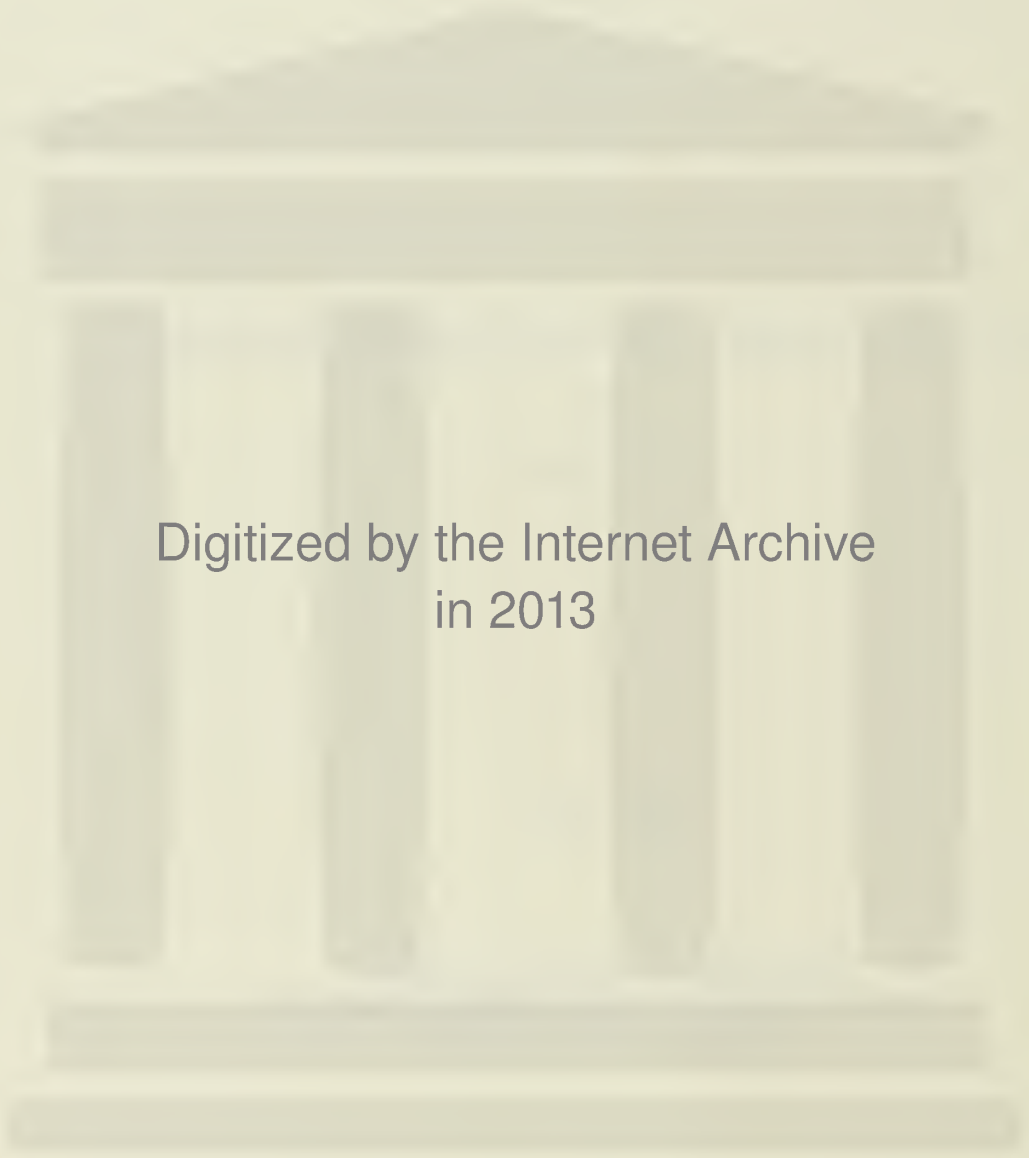
by

C. W. Gear

March 1979

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

Supported in part by the U.S. Department of Energy, Grant US ENERGY/EY-76-S-02-2383.



# Initial Value Problems: Practical Theoretical Developments

C. W. Gear

Department of Computer Science

University of Illinois at Urbana-Champaign

Urbana, IL 61801, USA

## Abstract

This paper surveys several recent developments in the theory of multistep initial value solvers that are useful for the implementation of practical codes. These developments are concerned with stability of variable step/order methods, starting at a high order (and hence, with a large step), and the integration of problems with rapidly oscillating solutions.

## Introduction

The work surveyed here is aimed at improving the efficiency of multistep, variable order, variable step codes for initial value problems. A code will be efficient if it can use a large step size, and can rapidly increase the step when a change in the nature of the problem warrants it. One of the limits to step size changing is the problem of stability, so it is important to use as weak a restriction on step size as possible consistent with a guarantee of stability. Further, the restriction should be easy to implement so that it does not contribute to the overhead. Recent results have shown that a very simple restriction that bounds the ratio of adjacent steps above and below by constants specific to the class of methods used is

sufficient to guarantee stability.

Comparisons of codes by numerous workers, including Hull et al [8] have shown that variable-order, multistep methods are among the most competitive for initial value problems except in some circumstances. The exceptions occur when a function evaluation is extremely inexpensive, in which case the increased overhead of multistep methods makes them less attractive than Runge-Kutta codes, or when a problem has a large number of discontinuities which cause relatively expensive restarts. A technique has been developed which allows a multistep method to start (or restart) at a high order and large step size, and yet uses relatively few function evaluations. It is a Runge-Kutta-like technique, and will, for example, let a method start at fourth or fifth order after only 6 function evaluations. It also provides information that allows a reasonable estimate of the initial step to be made. Preliminary tests indicate that it is efficient and allows multistep methods to be used in several cases where previously they were not competitive. The starting technique has the additional advantage that it can be inserted into many of the popular existing codes with very little difficulty, as it can be used to calculate whatever type of information is stored as the history of the problem, whether past function values, past derivative values, divided or backward differences, or high-order derivatives.

There are still some problems whose numerical computation is so difficult as to be effectively impossible. Among this class of problems were the problems with highly oscillatory solutions. Some authors [11] have termed these "stiff oscillatory" problems because the difficulty can occur when there are imaginary eigenvalues large compared to the interval of integration. However, I think this is an unfortunate misnomer as the basic difficulty in the conventional stiff problem is to ignore the effect of components not present in the solution. In my dictionary, a problem is stiff only when an eigenvalue has a large negative real part relative to the activity in the solution. If we have a problem with a large imaginary eigenvalue, and the solution contains no component corresponding to that eigenvalue, then we might consider it as a stiff problem, and some of the techniques for stiff problems will work (but often not the backward differentiation formulae in DIFSUB). However, if the eigenvalue is almost pure imaginary so that the component has no damping, the probability is that the oscillating component will be present in the solution. If the problem is linear, it does no harm to ignore that component, but in the class of the much more interesting non-linear problems, the behaviour of the oscillation must be considered. Thus, even if the oscillation is due to an imaginary eigenvalue, we need methods to take account of the oscillation in general. Furthermore, many oscillations are non-linear in nature, for example the relaxation oscillation of the van der Pol equation

$$y'' + k(y^2 - 1)y' + y = 0, \quad k > 0$$

in which the eigenvalues switch rapidly from the negative to the positive half plane. The computational objective may be one of several: for example, to determine the nature of the oscillation after an arbitrarily long time (sometimes called the "periodic steady state problem"), to find the phase of the oscillation after a long time, to determine the amplitude of the oscillation, or to determine the behaviour of the underlying "non-oscillatory" components. Only in the latter case are we interested in ignoring the oscillatory components altogether, and it appears that even then, their effect must be considered if the problem is non linear. The PhD thesis of a recent Illinois graduate, L. Petzold [12], gives extensions of some methods developed by various workers, including Graff and Bettis [6], Mace and Thomas [10], and others, and studies their accuracy. These methods are relatively easy to embed in standard codes such as DIFSUB.

## 1 Stability

It has been known for some time that arbitrary step changes and/or order changes can make some multistep methods unstable. One of the first positive results in this direction was due to Piotrowski [13] who showed that Adams methods are unconditionally stable if implemented in a variable step form. Later, Tu [14] showed that the conditions on programs using interpolation techniques for changing the step size have to be more strict,

at least for Admas methods, as there exist sequences of steps for which the three-step Adams Bashforth Moulton method is unstable. Jackson and Skeel [9] investigated these methods thoroughly and showed that, for any strongly stable multistep method using the interpolation scheme for step changing, there exist constants $0 < a \leq b < 1 < B$ such that the program is stable if the ratio of adjacent step sizes $h_{n-1}$ and $h_n$ is such that either

$$b < h_n/h_{n-1} < B$$

or

$$0 < h_n/h_{n-1} < a$$

This means that either the rate of change of step size is restricted, or the step is reduced radically. This is a very practical restriction as it is easy and inexpensive to implement, and it allows a very small step to be used when a sharp reduction is necessary because of a sudden change in the behavior of the solution. All that is necessary is to check the step ratio recommended by your favorite error control scheme, and reduce it to B if it is larger than B, or to reduce it to a if it is between a and b. More recently, Gear [4] has shown that the same condition is sufficient for multistep methods to be stable. While the underlying reason is, in one sense, quite technical, it admits a very simple intuitive explanation which is given in [4].

## 2 Starting multistep methods.

Most current multistep codes are variable order, and this feature allows them to start at a low order corresponding to a one-step method. This is typically first or second order, and implies a fairly small time step for modest accuracy. Consequently, the initial steps are fairly expensive in that they use a large number of function evaluations before the order reaches a level at which the step is close to maximum. This is not a serious drawback when a problem has to be integrated over a considerable range, since the number of function evaluations in the start-up is far less than the number used over the rest of the range. However, if the problem has frequent discontinuities, a multistep method must restart after every discontinuity. In this case, a Runge-Kutta method is frequently preferable because, as long as the discontinuities are at mesh points, a one-step method is unaffected by discontinuities in derivatives. A recently developed technique given in Gear [5] overcomes this problem by using a Runge-Kutta like starter to generate values for a multistep method. To illustrate the idea, let us consider the question of finding a set of values for starting a 4-th or 5-th order in which the predictor is fourth order. This requires a 4-step method if we want strong stability. At the start, or after a discontinuity, we have a single starting value, so the problem is to generate three additional values with appropriate accuracy. Since we are going to use a variable order method subsequently, it does not

make much sense to talk about order as a measure of accuracy, but unfortunately nobody has offered an alternative (and many of us have pondered this problem for some time). Therefore, we will ask that the three additional values be generated to fourth order accuracy. (Why not fifth order? -- since restarting occurs a fixed number of times, the contribution of a fourth order error at each restart is a global contribution of fourth order, so four seems a reasonable choice.) Given the equation $y' = f(y,t)$ and a single value of the pair $(y,t)$, it is clear that the first thing we do has to be an evaluation of $f(y,t)$. Consequently, if we somehow calculate y to fourth order accuracy at three additional points, we will have five fourth-order accurate values available, y and $y'$ at the initial point and three additional values of y. These uniquely define a fourth-order polynomial taking the same values. Therefore, the problem is to determine the coefficients of this polynomial, or any set of five values that uniquely determine them. The approach taken in [5] is to ask about the existence of Runge-Kutta-like processes that will generate these values to fourth-order accuracy. To be precise, we ask if it is possible, using a q-stage explicit Runge-Kutta process, to calculate the values of the scaled derivatives

$$[y_0,\ hy_0',h^2y_0^{(2)},\ \ldots\ ,\ h^py_0^{(p)}\ ]$$

to p-th order accuracy. The process used is

$$k_i = hf\left(y + \sum_{j=1}^{i-1} \beta_{ij} k_j\right) \quad i = 1,\ldots,q$$

$$h^s y^{(s)} \simeq \sum_{j=1}^{q} \gamma_{sj} k_j \quad s = 1,\ldots,p$$

Although this process ressembles the Runge-Kutta-type methods that generate several approximations of different orders (see, for example, Bettis [1], and Verner [15]), it is not the same, as this is equivalent to generating several output values at the same order. Reference [5] shows that the minimum value of q for p = 1 to 4 is

| p | minimum q |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 4 |
| 4 | 6 |

The minimum value of q for other p is not known, although it has been shown that the minimum is no larger than $1 + p(p-1)/2$. Of course, the presentation of a set of formula does not necessarily lead to a practical algorithm; there is still the very important question of step selection, and this is now a double problem: what step to use in the Runge-Kutta process, and what step to use in the first multistep step. Note that the

Runge-Kutta process does not actually take a step as we have presented it. If its output is used to start a code that uses a Nordsieck vector, this remains true. If, on the other hand, we use it with a code that retains past function or derivative values, we must replace the linear combinations of the k's with other linear combinations that give the required values of y or hy' at a set of points. We could choose either points forward of the starting point, or points behind the starting point. My preference is the latter. Then, we do not have to be concerned about the Runge-Kutta starting process introducing error directly into the solution, but only with the propogation of errors it introduces in the additional values. The step size for the Runge-Kutta process must be chosen small enough that this propogated error is reasonable, but large enough so that roundoff errors do not overwhelm other errors. If we use a fourth order process, for example, then we can determine if the step size is within these (hopefully generous) bounds by examining the values of the scaled derivatives calculated. While it is true that the error is an unknown fifth-order term, we can use the popular approach of controlling the size of the fourth-order terms, coupled either with a small prayer or a generous safety factor, determined by your philosophical predilections. At first sight it seems that the scaled fourth derivative times an error coefficient should be less than the error tolerance $\varepsilon$ (scaled by the function value for relative errors). However, this is really the error control we would apply if we were using the RK

information to take a step. Suppose the h we use in the RK step is larger than this. Clearly we will reduce the step in the multistep method so that its error is within tolerance. When we do this, we will rescale the scaled derivatives calculated in the starter by powers of the step size ratio. The second scaled derivative is the first one we calculate that has errors (the first derivative has no truncation error). This is scaled by the square of the step ratio. After the scaling, we would like the fifth-order error present in that term to be less than $\varepsilon$. However, we agreed to estimate the error by looking at the fourth order derivative. Bringing this together, we get:

Original scaled second derivative is

$$h^2 y^{(2)} + \text{error}$$

Assume that the error is given by

$$Ch^4 y^{(4)}$$

Error in second derivative after scaling by $(h_{new}/h)^2$ is

$$E = Ch^2 y^{(4)} (h_{new})^2$$

If the error control in the first step taken is such that

$$C_4 h_{new}^4 y^{(4)} = \varepsilon$$

we can calculate $h_{new}$, insert it in the previous expression to get

$$E = Ch^2 y^{(4)} \left[ \frac{\varepsilon}{C_4 y^{(4)}} \right]^{1/2}$$

If we want to control E so that

$$KE \leq \varepsilon$$

we find that

$$h^4 y^{(4)} \leq \frac{C_4 \varepsilon}{K^2 C^2}$$

This is the test that must be applied to see whether to reject the Runge-Kutta starter. (We have assumed that the error in the rescaled third scaled derivative will be smaller and can be ignored.) The fourth scaled derivative should also be about an order of magnitude larger than u, the unit round-off error (also scaled by the function value if necessary). Thus we see that the test of acceptability for the Runge-Kutta starter is simply a check on the size of the fourth derivative (or p-th derivative, in general. However, we have to be beware of the possibility of zero p-th

order derivatives. These are not so uncommon at initial values, so it is important to consider the value of all derivatives up to the p-th in estimating the error. Our approach is to assume that the behaviour of the q-th derivative is

$$||y^{(q)}|| \simeq \lambda^q ||y|| \tag{1}$$

Using this, we calculate

$$\frac{||y^{(p)}||}{||y||} \simeq \left[\frac{1}{p}\sum_{j=1}^{p}\left[\frac{||y^{(j)}||}{||y||}\right]^{1/j}\right]^p \tag{2}$$

as an approximation to the p-th derivative relative to y. (If $||y||$ is less than 1, we replace $||y||$ with 1.)

Once the Runge-Kutta starter has been performed with a satisfactory step we can choose a step for the first application of a multistep method. Once again, we use the estimated p-th derivative to estimate the error in a (p-1)st order method, even though we may use a p-th or (p+1)st order method. If the error in the (p-1)st order version of the method used is

$$C_p h^p y^{(p)}$$

we choose an initial h based on

$$h_{new} = \left| \frac{\varepsilon \, ||y||}{C_p \, ||y^{(p)}||} \right|^{1/p}$$

We have also tried using

$$h_{new} = \left| \frac{\varepsilon \, ||y||}{C_q \, ||y^{(q)}||} \right|^{1/q}$$

with $q = p+1$ and $p+2$, where our estimate of the q-th derivatives of y is based on the approximation in eq. (1) and the estimate in eq. (2). This yields the estimate

$$||y^{(q)}|| \simeq ||y|| \left[ \frac{1}{p} \sum_{j=1}^{p} \left[ \frac{||y^{(j)}||}{||y||} \right]^{1/j} \right]^{q}$$

All this does is to change the dependence of the initial step on $\varepsilon$ and to change the coefficient used.

## 2.1 Detecting Discontinuities

If an automatic step control integrator is given no further information about discontinuities, the step control mechanism can be very inefficient in the neighborhood of a discontinuity. This occurs because an attempted step that straddles a discontinuity yields a very large error estimate,

usually causing the code to reject the step and reduce the stepsize sharply so that the step not only no longer straddles the discontinuity but often so that it falls far short of the discontinuity. Consequently, several small steps are taken until the step is once again increased to a reasonable size so that an attempted step again straddles the discontinuity. This process is repeated with a large cost in function evaluations. Codes have used a number of techniques to reduce the problem. DIFSUB, an early code, reduces the order to one and restarts when this type of difficulty is first encountered, and then quits if it occurs three times in succession--a simple but not too useful solution. Some codes resort to binary division of the interval when the error tests are inconsistent. (The error has a known asymptotic behavior if no discontinuities are present; it is possible to check for gross deviation from such behavior as an indication of discontinuities.) Binary division may be as good as any strategy if no further information about the discontinuity is available. However, if we know exactly when each discontinuity occurs, it is far more efficient to integrate exactly to the discontinuity and then restart.

Discontinuities can arise from two sources: the driving (t-dependent) terms and the dependent variables. An example of a driving term discontinuity occurs in smog simulation. The sun's energy input is a time-dependent term that has two discontinuities each 24 hours as the sun rises and sets. Dependent variable discontinuities frequently arise in

simulation. For example, a relay leads to equations of the form

$$\text{if } |IR| > I_{critical} \text{ then switch} \leftarrow \text{on}$$

$$\text{if } |IR| < I_{min} \text{ then switch} \leftarrow \text{off}$$

$$\text{if switch} = \text{on then } V = 0$$

$$\text{else } I = 0$$

where IR is the current in the relay coil, and V and I are the voltages across and the current through the relay contracts. This means that as IR (which is a function of the dependent variables) increases to the point it passes $I_{critical}$, the relay turn on current, the coefficients in the differential equations change discontinuously. A similar problem arises in some mathematical models that, although infinitely differentiable, exhibit such sudden changes they are indistinguishable from discontinuities. For example, a diode is sometimes modeled by an exponential function such as

$$I = c(e^{40V} - I)$$

where c is the reverse leakage current. The constant 40, or some large number, causes a behavior similar to that shown in Figure 1. In such cases either the model should be changed to a discontinuous model (e.g., a piecewise linear function) or detection schemes which indicate when the nature of the model changes radically should be added.

The basic problem is to detect when a discontinuity occurs in time so that when a step is about to straddle it, the stepsize can be reduced. This is trivial for time-dependent discontinuities, so let us consider the problem of functions of dependent variables. Let us suppose that a discontinuity occurs when an expression e(y) crosses zero. Since y is a function t, we can evaluate e(y(t)) at each integration step. We could use a set of past values $e_{n-i}$, i = 0,..., k from a multistep method to estimate the time at which the next zero crossing will occur and reduce the step of that time is within the next interval. However, that is time-consuming because of an inverse interpolation in each interval and error prone because extrapolation gives larger errors than interpolation. Therefore, we would like to be able to take the next step and then interpolate. This is possible only if there is no discontinuity in the tep. The technique employed by many people (see Carver [2] is to delay changing the "switch" until the completion of a step. Thus, if e(y) is an expression such that a swtich is to change when e(y) changes from negative to positive, e(y) is evaluated only at the end of each step. If e(y) remains negative, the integration proceeds normally. However, if the value of e(y) is found to be positive at the end of a step, it indicates a discontinuity somewhere in that step. Inverse interpolation can then be employed to determine the t value at which e(y) is zero. Interpolation can then be employed to calculate the dependent variables, although it is probably better to

integrate from the last value of t, and posible iterate this procedure one time.

## 3 Oscillating problems

At the outset, we must distinguish between two types of problems; those whose oscillation is due to a forcing term (time dependent term), and those whose oscillation is "internal", that is, due to a linear or nonlinear oscillator in the equations. Initially we will discuss the latter, and later indicate extensions to the former case. We expect the solution to have the behaviour sketched in Figure 2, and our goal is to determine the nature of the solution after a very long time; that is, after a large number of cycles. First, we must be more precise about what we want to know about the solution. Chances are that we either want to know the form of the oscillation, or the "envelope" of the solution; that is, the curves $z_H$ and $z_L$ in Figure 2.

Unfortunately, there is no such thing as an envelope of a single curve, and the solution of an ODE with an initial value is a single curve. However, it is quite obvious to the casual observer what we mean by the envelope in the case of a very rapidly oscillating solution. Our objective was to develop a method which would allow us to calculate a smooth curve that was the "envelope" so that we could use large step sizes. Therefore, we defined a (hopefully) smooth curve, called a quasi envelope, and have

derived approximate differential equations for it. The essence of the method is to integrate this approximate differential equation using large time steps.

The quasi envelope is illustrated in Figure 3. Suppose the "period" of the oscillation is T. (We will discuss how this is defined later, and indicate how a "varying period" is handled. For the moment assume that we have been given the value of T.) If the initial value of y at $t_0$ is $y_0$, we choose a function z(t) for $t_0 \leq t \leq t_0 + T$ such that z and y agree at $t_0$ and $t_0 + T$. Between these values z(t) should be smooth. Now we extend z(t) to all $t \geq t_0 + T$ by defining $z(t + T)$ to be the value of the solution at $t + T$ of $y' = f(y,t)$ with initial values t, z(t). Hence, z(t) is a collection of points such that if y and z agree at time t, they agree at times $t + kT$ for all integer k. The smoothness of z(t) depends in a complex way on the smoothness of the initial section in [0,T], but this is only an impediment to the theoretical treatment of truncation, not to the practical algorithm.

Formally, we define the quasi envelope by

$$z(t + T) = z(t) + Tg(z,t) \tag{3}$$

where

$$g(z,t) = \frac{1}{T}[y(t + T, t) - y(t, t)] \tag{4}$$

and $y(t, \tau)$ is the solution of

$$\frac{d}{dt} y(t, \tau) = f(y(t,\tau), t), \quad y(\tau,\tau) = z(\tau)$$

An example of y is the dashed curve in Figure 3. It "moves the solution forward one cycle." At first sight, many find the quasi envelope confusing. "What if we choose an initial $z(t)$, $0 \leq t < T$ such that we are in the middle rather than on the peaks? Then we will have no idea of the amplitude." It must be remembered that Figure 3 is confusing as it shows only one dimension of at least a two-dimensional space of solutions y. (An autonomous problem must be multidimensional to oscillate.) Hence, the quasi envelope is "on the side" of the oscillation. If the solution is truly periodic, it is a fixed point in phase space.

Notice that $g(z,t)$ in eq. (4) is a function that can be calculated by an integration of the original ODE over one period. We refer to this as the "inner integration." Now we want to find a technique to compute z from eq. (3) given g. The technique is essentially that developed by other authors (see [6], [10], for example). The objective is to evaluate g periodically, say at points $t_N = NH$ where $H >> T$ and use this information to approximate z. The standard formulas are most easily developed using operator calculus. Let D be the differentiation operator, and let

$\nabla v(t) = v(t) - v(t-H)$. Let $z_N$ be the computed approximation to $z(t_N)$.

From eq. (3) we have

$$z_N = \left[\frac{e^{TD}-I}{T}\right]^{-1} g_N \tag{5}$$

Hence

$$z_N - z_{N-1} = \nabla\left[\frac{e^{TD} - I}{T}\right]^{-1} g_N$$

$$= \nabla\left[\frac{\exp[-\frac{T}{H}\log(I-\nabla) - I}{T}\right] g_N$$

$$= H\{[I - \frac{1}{2}\nabla - \frac{1}{12}\nabla^2 - \ldots]$$

$$-\frac{1T}{2H}\nabla - \frac{1}{12}\left|\frac{T}{H}\right|^2\nabla^2 + \ldots \tag{6}$$

The first term in square brackets corresponds to the Adams Moulton formula. The remaining terms ar correction terms which are small if T/H is small. Graff [7] has called these "generalized Adams" methods. The above is an implicit method because $g_N$ depends on $z_N$. We could equally well have arrived at an implicit generalization of Adams Bashforth methods. The two together give a standard predictor/corrector pair which can be employed in

the usual way. Alternatively, we can use eq. (5) to express $g_N$ in terms of $z_N$ and its backward differences to get an implicit generalized backward differentiation formula.

When we use a formula such as that embodied in eq. (6) we discard high powers of $\nabla$. This leaves a multistep formula of the form

$$\sum_{i=0}^{k} [\alpha_i(r) z_{N-i} + H\beta_i(r) g_{N-i}]$$

where $r = T/H$ and $\alpha_i$ and $\beta_i$ are polynomials of degree k in r.

A practical code must vary its stepsize H (and the order). At first sight this looks to be a complex mess. However, Petzold [12] has shown that interpolatory stepsize changing is easy if an analogue of the Nordsieck vector is used to represent the history of the solution. Instead of storing $[z, Hz', H^2z^{(2)}/2, \ldots, H^kz^{(k)}/k!]^*$ she stores

$$\underline{a}_N = [z_N, H\nabla_T z_N, H^2\nabla_T z_N'/2, \ldots, H^k\nabla_T z_N^{(k-1)}/k!]^*$$

where

$$\nabla_T z(t) = \frac{1}{T}[z(t+H) - z(t)]$$

When this form is used, the predictor step takes the form

$$\underline{a}_{N,(0)} = A(r)\underline{a}_{N-1}$$

where A(r) is the Pascal triangle matrix in all but the first row where it is a polynomial in r = T/H. The corrector takes the form

$$\underline{a}_{N,(m+1)} = \underline{a}_{N,(m)} + \underline{\gamma}(r)F(\underline{a}_{N,(m)})$$

where $\underline{\gamma}$ is independent of r in all but its first component. Those of you familiar with the Nordsieck scheme (see [3]) will recognize that this is a minor change to that scheme. In fact, the stepand order changing algorithms are identical, so that only minor changes were needed to DIFSUB (see [3], chapter 9) to adapt it.

Variable Period

A variable period T(t) is handled by a simple change of independent variable. Assume we know T(t). We introduce a new independent variable s such that

$$s(t+T(t)) - s(t) = \tau, \quad s(0) = 0$$

where $\tau$ is the constant period in the new variable. However, what we need to calculate is t(s) so that we can evaluate g(z(t(s)),t(s)). We have

$$t(s+\tau) - t(s) = T(t(s)), \quad t(0) = 0$$

Rewriting this and eq. (3) we get

$$z(t(s+\tau)) = z(t(s)) + \tau\left[\frac{T(t(s))}{\tau} g(z,t(s))\right]$$

$$t(s+\tau) = t(s) + \tau\left[\frac{T(t(s))}{\tau}\right]$$

The variable t is handled as an additional dependent variable. We solve both of these equations using the constant period formulas given above with period $\tau$, where g is replaced by $T(t(s))g(z,t(s))/\tau$ when computing z, and by $T(t(s))/\tau$ when computing t.

Period Determination

The period is not defined for a non-periodic function, but we should notice that the preceding discussion is equally applicable to any function and any T(t), whether periodic, "nearly periodic" (whatever that means), or arbitrary. If y(t) is highly oscillatory and T(t) is in no sense a "period," the function z(t) will have components very similar to the high frequency components of y, so H must be small for accuracy and nothing is gained. However, if we can find a T(t) such that z(t) is slowly varying, we can hope to use a large H.

Let us suppose that the solution y has the form

$$y(t) = w(t) + p(t)$$

where w is periodic and p(t) is slowly varying. Define the period of y to be the value of T such that

$$I(T) = \min_{p \in P} \int_0^T [y(t+T+s) - p(t+T+s) - y(t+s) + p(t+s)]^2 ds \tag{7}$$

is minimum for $T \geq T_0$, where P is some finite dimensional class of functions (e.g. polynomials of degree m). If the assumption on y is valid, and if p is in P, then I(T) is zero for T equal to multiples of the period of w. (We constrain $T \geq T_0$ to avoid the undesirable solution T = 0.) Given y and T, the computation of p from (7) is straightforward because it is a quadratic minimization problem in p. Minimizing with respect to T is more difficult, so we have used a few steps of an iteration to find a zero of its derivative. In practice, we have found that we can take p = 0 because, for problems of interest, p is so slowly varying compared to w it can be ignored.

The algorithm for solving $y' = f(y,t)$ now takes the following general form:

1. Start with a small H and a one-step outer integration. An initial estimate of T must be provided. $y_0$ is given.

2. Evaluate $T_0$, $g_0$ (see "subroutine" below). Set

$N = 0$, $z_0 = y_0$, $t_0 = 0$.

3. Predict $t_{N+1}$, $z_{N+1}$ using outer integration.

4. Evaluate $T_{N+1}(z_{N+1})$, $g_{N+1}(z_{N+1})$ (see subroutine).

5. Correct $t_{N+1}$, $z_{N+1}$ using outer integration.

6. Repeat steps 4 and 5 as desired.

7. Perform a final evaluation of $T_{N+1}$ and $g_{N+1}$ if desired.

8. Increment N. Select new step and order. Repeat steps 3 to 8 as necessary.

Subroutine to evaluate T and g.

S1. Set $y = z$

S2. Integrate $y' = f(y,t)$ for two cycles using inner integration

S3. Compute new T with an iteration applied to $I'(T) = 0$ (see eq. (7))

S4. Compute g from y

Nonautonomous Systems

If the problem has high-frequency driving terms, then it is important to

keep phase information about the solution. Referring to Figure 3, note that the dashed line is not a part of the solution of the original problem, but, for an autonomous system, it is essentially a phase shift of a nearby part of the solution. However, if we advance t by multiples of the period H only, we will remain in phase. All this means is that we restrict H to integer multiples of T. We call this the synchronized mode. Fortunately, when there are oscillatory forcing functions, we usually know the period exactly, so it is not difficult to stay synchronized. Tests on a few simple problems reported in [12] indicate speed improvements of twenty- and forty-to-one for a nonlinear autonomous problem (lightly damped large amplitude pendulum) and a linear nonautonomous problem (linear oscillator with high frequency forcing function). The error in the generalized methods was slightly better than that of DIFSUB.

Error Analysis

The interested reader is referred to Petzold's thesis [12] for details. Essentially, the analysis shows that the inner integration error is the critical one. It is easy to pick parameters so that the contribution from the outer integration is small. The contribution from the inner integration will be about the same as if the inner integration were to be used over the full range. (This should not be viewed lightly. If the oscillation is very fast, it may be impossible to solve over the whole

range because of roundoff! The problem can be seen by examining g. We see from eq. (4) that as $T \geq 0$, errors in computing y over one cycle of the inner integration grow with $\frac{1}{T}$.)

## 4 Conclusion

There are many problems still to solve before the universal automatic integrator can be written. Among these are the development of reliable schemes for determining the nature of the equation and solution (stiff, oscillatory, etc) in order to select a method. Also the control of global error in an efficient manner presents a challenge that is likely to be frustrated by the lack of theory. The asymptotic dependence of the global error on the local error control parameter needs to expressed in a form that is useful. Unfortunately, it appears that with most popular local error control schemes it does not have a useful form. Also, there is a lack of theory for "large" errors, ones for which asymptotic analysis is inappropriate. In practice, we compute with a step size that is determined once the problem and error control are specified. Hence, we are not interested in the behaviour of the error as the step decreases, but in the actual error committed. We would like to minimize that over a choice of methods (or rather, maximize h for a constant error). Thus, we are not the least interested in the order of the method, nor its convergence. A method with zero order (and hence not convergent) might be the best method for the

particular error and problem. We have no theory to build on these ideas.

Bibliography

[1] Bettis D. G. Efficient embedded Runge-Kutta methods, in Numerical Treatment of Differential Equations, eds Burlirsch, Grigorieff, and Schroder, Springer-Verlag, Berlin, 1976

[2] Carver M. B. Efficient handling of discontinuities and time delays in ordinary differential equations, Proceedings of Simulation '77 Symposium, Montreux, Switzerland, ed M. Hamza, Acta Press, Zurich 1977

[3] Gear C. W. Numerical Initial Value Problems in Ordinary Differential Equations, New Jersey: Prentice-Hall 1977

[4] Gear C. Stability of variable step methods for ordinary differential equations, Report #931, Dept. of Computer Science, University of Illinois at Urbana-Champaign, July, 1978

[5] Gear C. W. Runge-Kutta starters for multistep methods, Report #78-938, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 1978

[6] Graff O. F. Bettis D. G. Modified multirevolution integration methods for satellite orbit calculation, Celestial Mechanics 11, pp 443-448, 1975

[7] Graff O. F. Methods of orbit computation with multirevolution steps.

Applied Mechanics Research Laboratory Report AMRL 1063, University of Texas, Austin 1977

[8] Hull T. E. Enright W. H. Fellen B. M. Sedgwick A. E. Comparing numerical methods for ordinary differential equations, SIAM Journ. Num. Anal. 9, pp 603-637, 1972.

[9] Jackson L. W. Skeel R. D. Convergence and stability of Nordsieck methods Technical Report #89, Department of Computer Science, University of Toronto, March 1976

[10] Mace D. Thomas L. H. An extrapolation method for stepping the calculations of the orbit of an artificial satellite several revolutions ahead at a time, Astronomical Journal 65 #1280, June 1960

[11] Miranker W. L. Wahba G. An averaging method for the stiff, highly oscillatory problem, Math. Comp. 30, pp 383-399, 1976

[12] Petzold L. An efficient numerical method for highly oscillatory differential equations, Report #933, Dept. of Computer Science, University of Illinois at Urbana-Champaign, 1978

[13] Piotrowski P. Stability, consistency, and convergence of variable K-step methods for numerical integration of systems of ordinary differential equations, Conference on the Numerical Solution of Differential Equations,

ed. J. Ll. Morris, Springer, Berlin, pp 221-227, 1969

[14] Tu K. W. Stability and convergence of general multistep and multivalue methods with variable step size, Doctoral thesis, Report #526, Department of Computer Science, University of Illinois at Urbana-Champaign, July, 1972

[15] Verner J. H. Explicit Runge-Kutta methods with estimates of the local truncation error, SIAM J. Num. Anal. 15, #6, pp 772-790, Aug 1978

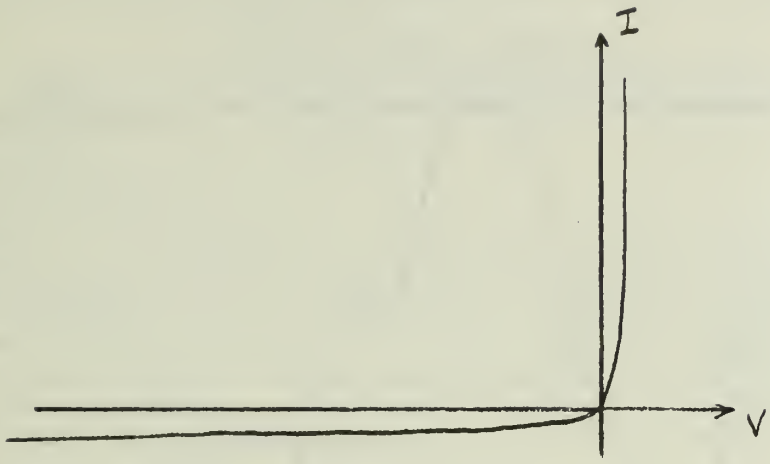



Figure 1. Nearly discontinuous diode current-voltage relationship.

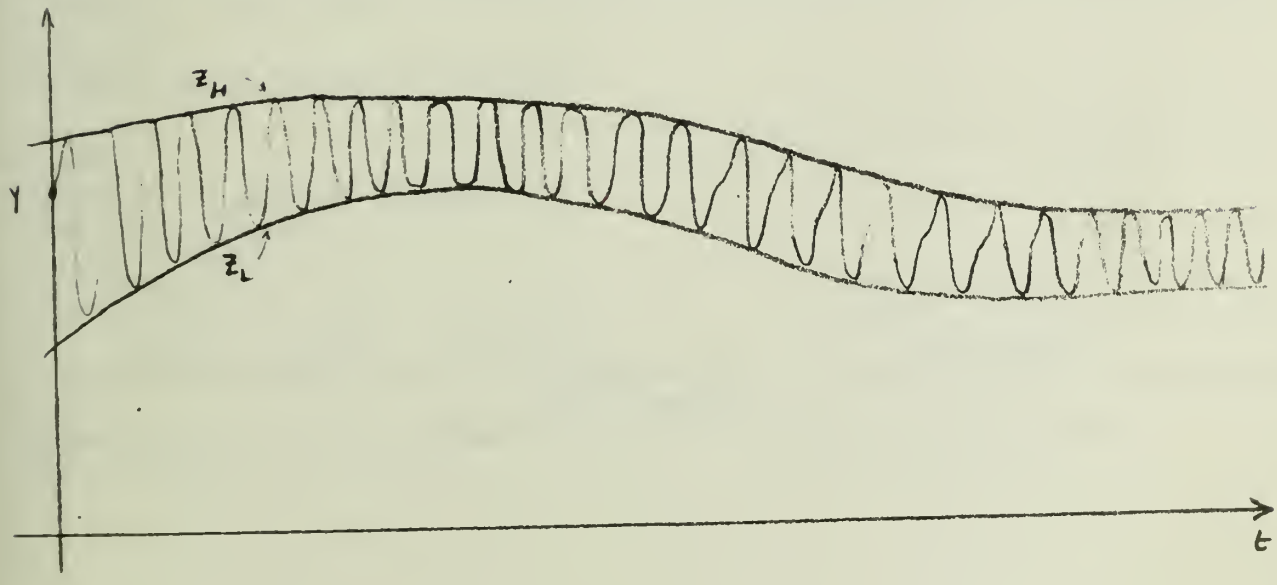



Figure 2. Envelope of a solution.

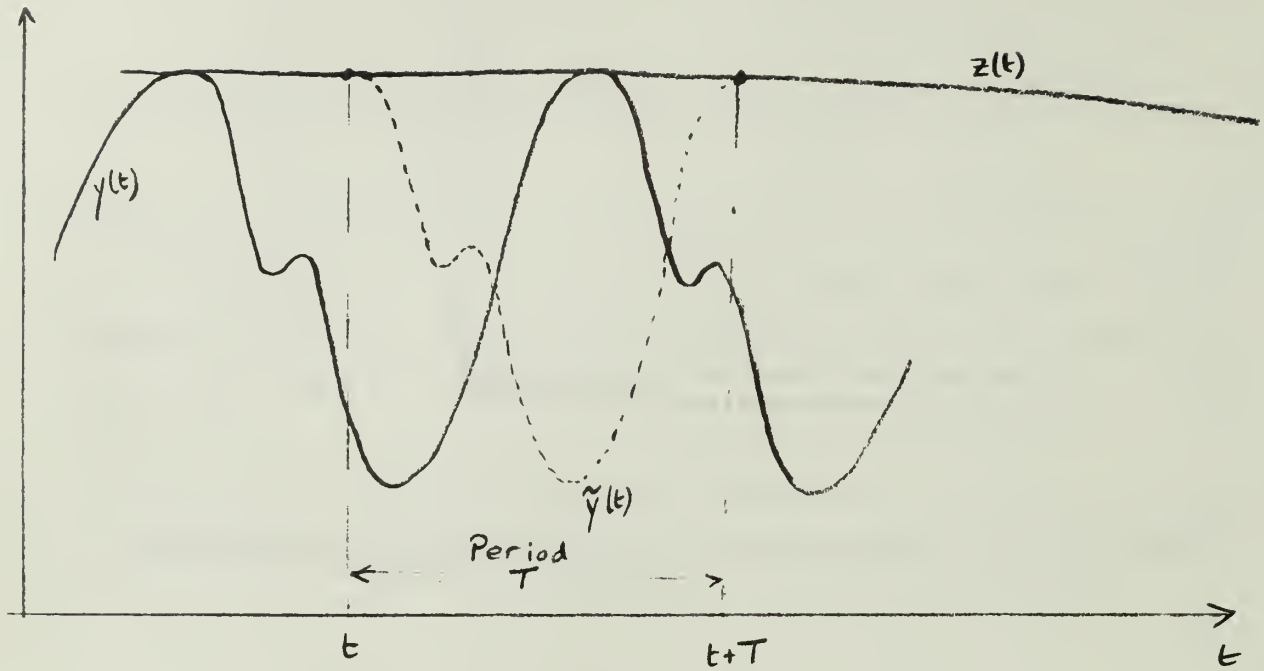



Figure 3. Quasi Envelope.

Form AEC-427
(6/68)
AECM 3201

# U. S. ATOMIC ENERGY COMMISSION
# UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

*( See Instructions on Reverse Side )*

1 AEC REPORT NO.

COO-2383-0057

2. TITLE

INITIAL VALUE PROBLEMS:
PRACTICAL THEORETICAL DEVELOPMENTS

3. TYPE OF DOCUMENT (Check one):

- [X] a. Scientific and technical report
- [ ] b. Conference paper not to be published in a journal:
  - Title of conference _____
  - Date of conference _____
  - Exact location of conference _____
  - Sponsoring organization _____
- [ ] c. Other (Specify) _____

4 RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- [X] a. AEC's normal announcement and distribution procedures may be followed.
- [ ] b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
- [ ] c. Make no announcement or distrubution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

C. W. Gear
Professor and Principal Investigator

Organization

Department of Computer Science
University of Illinois U-C
Urbana, IL 61801

Signature

C. W. Gear

Data

March 1979

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

- [ ] a. AEC patent clearance has been granted by responsible AEC patent group.
- [ ] b. Report has been sent to responsible AEC patent group for clearance.
- [ ] c. Patent clearance not required.

| BIBLIOGRAPHIC DATA SHEET | 1. Report No. UIUCDCS-R-79-962 | 2. | 3. Recipient's Accession No. |
|---|---|---|---|

4. Title and Subtitle

INITIAL VALUE PROBLEMS: PRACTICAL THEORETICAL DEVELOPMENTS

5. Report Date
March 1979

6.


7. Author(s)
C. W. Gear


8. Performing Organization Rept. No. UIUCDCS-R-79-962


9. Performing Organization Name and Address

Department of Computer Science
University of Illinois U-C
Urbana, IL 61801


10. Project/Task/Work Unit No.

11. Contract/Grant No.
ENERGY/EY-76-S-02-2383

12. Sponsoring Organization Name and Address

U.S. Department of Energy
Chicago Operations Office
9800 South Cass Avenue
Argonne, IL 60439

13. Type of Report & Period Covered
Presented at IMA Conf. Manchester, England 1978

14.

15. Supplementary Notes

16. Abstracts


This paper surveys several recent developments in the theory of multistep initial value solvers that are useful for the implementation of practical codes. These developments are concerned with stability of variable step/order methods, starting at a high order (and hence, with a large step), and the integration of problems with rapidly oscillating solutions. This paper was presented at the IMA Conference on the Numerical Solution of Differential Equations, Manchester, England, December 19, 1978.


17. Key Words and Document Analysis. 17a. Descriptors


differential equations
oscillatory solutions
discontinuities
Runge-Kutta
stability


17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement
unlimited

19. Security Class (This Report)
UNCLASSIFIED

20. Security Class (This Page)
UNCLASSIFIED

21. No. of Pages

22. Price